# Oque é uma activity?

É um componente do android que representa a tela do nosso aplicativo, onde vamos ter: textos, botões, menus, listas entre outros.

# Uma activity ela é composta por;

Uma class Kotlin e um arquivo XML onde vai ter a parte gráfica.

Os dois juntos vão formar uma acitivity

# Navegando entre activity

Antes de ir para a navegação entre as activity o ideal é entender oque é uma **PILHA** de activity

Uma pilha de activity é; conforme você vai abrindo, uma vai ficando sobre a outra, então o ideal é ir fechando-as para não ocupar muito espaço na memória.

Vamos imaginar que temos 2 activity e você abriu a segunda, então você não precisa voltar para a primeira, você pode apenas fechar a segunda e automaticamente vai retornar para a primeira.

# Como abrir uma nova activity

Para abrir uma nova activity vamos utilizar o método startActivity

Agora para abri precisamos utilizar como parâmetro o objeto **intent**

**startActivity(intent)**

Intent = intenção || (resolve ou faz determinada coisa)

startAcitivity(intente(**contexto**, classQueVaiSerAberta))

**contexto** = contexto que será executado

## Oque é o CONTEXTO

É um ponto de acesso para informações globais do seu App.

Vamos imaginar aqui em casa, tem várias janelas.

- Aí eu falo Fernanda abra a janela (**intente = intenção**)

E aí eu preciso passar o **contexto**

Oque seria o contexto; seria qual janela Fernanda vai abrir, porque tem mais de uma janela

- Fernanda abra a janela(**intente**) da cozinha (**contexto**)

E aí eu passo dessa forma

startActivity(intente(contexto, janelaSegundoActivity::class.java))

## Outro exemplo

startActivity(intent(contexto, JanelaCaronaActivity::class.java))

Vamos imaginar que falo, abra a janela, mas não sabemos qual, então precisamos passar um pouco mais de contexto

Abra janela (intenção)?

Qual? (contexto)

Do carona (JanelaCaronaActivity::class.java)

Resumo a **intent** ela faz algo que você esta mandando, mas ela precisa do contexto para saber onde ela vai fazer.

Então no exemplo de cima a **intent** esta abrindo uma nova Activity, mas ela precisa saber qual (**contexto**) ela vai abrir, ai você usa o uma virgula e passa a activity que precisa ser aberta.

## Abrindo nova Activity na pratica.

class MainActivity : AppCompatActivity() {  
  
 lateinit var btnAbrirSegundaTela : Button  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_main*)  
 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.*main*)) **{** v, insets **->** val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())  
 v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)  
 insets  
 **}** btnAbrirSegundaTela = findViewById(R.id.*button\_abrir*)  
 btnAbrirSegundaTela.setOnClickListener {  
 startActivity(Intent(this, DetalhesActivity::class.*java*))  
 }}  
  
}

Estamos adicionando uma variável que ainda não inicializamos, vamos inicializar depois **lateinit**

Agora estamos transformando a variável em um objeto do tipo **button\_abrir**

E depois estamos fazendo uma função lambda para quando o usuário clicar nesse botão ele abrir uma nova tela.

**setOnClickListener** = um ouvinte que fica esperando ser clicado para disparar um evento, nesse caso o evento vai ser abrir a segunda tela.

This = é para dizer qual contexto vai ser executado no nosso caso, pedimos para ele executar apenas o DetalhesActivity, mas poderíamos pedir para ele executar algo dentro do DetalhesActivity.

## Fechando uma activity

Aqui estamos fechando a segunda activity então ele vai voltar automaticamente para a tela anterior

Não faça isso para a primeira activity se não ele vai fechar o aplicativo.

class DetalhesActivity : AppCompatActivity() {  
  
 lateinit var btnFechar : Button  
  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_detalhes*)  
 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.*main*)) **{** v, insets **->** val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())  
 v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)  
 insets  
 **}** btnFechar = findViewById(R.id.*button\_fechar*)  
 btnFechar.setOnClickListener {  
 finish()  
 **}** }  
}

# Clico de vida de uma Activity

Primeiro começamos pelo onCreate (Onde tudo que estiver la vai ser feito primeiro)

Ele vai automaticamente configurar nosso layout

class DetalhesActivity : AppCompatActivity() {  
  
 lateinit var btnFechar : Button  
  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_detalhes*) < configurar nosso layout  
 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.*main*)) **{** v, insets **->** val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())  
 v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)  
 insets  
 **}**

btnFechar = findViewById(R.id.*button\_fechar*)

Configurar nosso evento de click no botão.  
 btnFechar.setOnClickListener **{** finish()  
 **}** }  
}Aqui conseguimos manipular elementos da interface.

Aqui no onCreate pode ser carregado o progressBar, ele vai ficar rodando ate ela carregar.

Em seguida é chamado o método **onStart**()

Imagina um jogo onde temos que pegar os dados do banco de dados, então é aqui que ele vai ficar carregando para a gente.

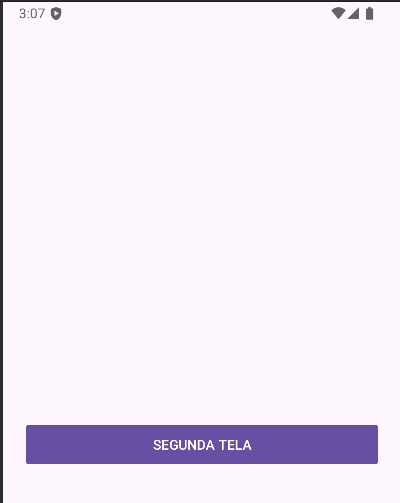
Depois o **onResume**()

Imagina assim : onCreate exibo o ícone de carregando

onStart : puxo os dados do banco

onResumo eu retiro o ícone de carregando

Após isso e renderizado a imagem no **Activity Running**



Agora vamos refazer todos esses métodos na pratica

## Refazendo o clico na pratica

package com.allephnogueira.aulaacitivityfragment  
  
import android.os.Bundle  
import android.util.Log  
import android.widget.Button  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.view.ViewCompat  
import androidx.core.view.WindowInsetsCompat  
  
class DetalhesActivity : AppCompatActivity() {  
  
 lateinit var btnFechar : Button  
  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 Log.i("clico\_vida", "onCreate")  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_detalhes*)  
 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.*main*)) **{** v, insets **->** val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())  
 v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)  
 insets  
 **}** btnFechar = findViewById(R.id.*button\_fechar*)  
 btnFechar.setOnClickListener **{** finish()  
 **}** }  
  
 override fun onStart() {  
 super.onStart()  
 //Carregar os dados que vem do servidor  
 //Imagina o alta pressão gnv, aqui é onde vamos carregar os dados do posto.  
 Log.i("cliclo\_vida", "onStart")  
 }  
  
 override fun onResume() {  
 super.onResume()  
 Log.i("clico\_vida", "onResume")  
 }  
  
 override fun onPause() {  
 super.onPause()  
 Log.i("clico\_vida", "onPause")  
 }  
  
 override fun onStop() {  
 super.onStop()  
 Log.i("clico\_vida", "onStop")  
 }  
  
 override fun onRestart() {  
 super.onRestart()  
 Log.i("clico\_vida", "onRestart")  
 }  
  
 override fun onDestroy() {  
 super.onDestroy()  
 Log.i("clico\_vida", "onDestroy")  
 }  
  
  
}

Qualquer duvida tem o material em PDF também.

# Passando dados de uma actvivity para outra.

Imagina o aplicativo Netflix, onde temos vários filmes, então quando clicamos em 1, vamos para a segunda tela, a tela de detalhes, mas detalhes de quem? Então precisamos passar as informações.

Vamos usar o

Intent.putExtra(“NOME\_FILME”, “Harry Potter”)

No primeiro parâmetro vamos passar o identificador

No segundo parâmetro vamos passar a informação que queremos utilizar.

Logo após fazer isso vamos chamar a classe, veja no código.

val intent = Intent(this, DetalhesActivity::class.*java*)  
  
btnAbrirSegundaTela = findViewById(R.id.*button\_abrir*)  
btnAbrirSegundaTela.setOnClickListener **{** // Vamos passar os dados para a proxima activity por parametro.  
 // Primeiro valor é o identificador do que voce quer passar  
 // Segundo é o dado  
  
   
 intent.putExtra("filme", "Harry Potter")  
 intent.putExtra("classificacao", 8)  
 intent.putExtra("avaliacoes", 300)  
 intent.putExtra("exibicao", false) // exibição nos cinemas?

startActivity(intent) // Estamos chamando o intente já configurado com o nome da classe.  
**}**

## Capturando na outra tela os dados que chegaram

class DetalhesActivity : AppCompatActivity() {  
  
 lateinit var btnFechar : Button  
 lateinit var textFilme : TextView // onde vamos colocar o nome do filme  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 Log.i("clico\_vida", "onCreate")  
 *enableEdgeToEdge*()  
 setContentView(R.layout.*activity\_detalhes*)  
 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.*main*)) **{** v, insets **->** val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())  
 v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)  
 insets  
 **}** btnFechar = findViewById(R.id.*button\_fechar*)  
 textFilme = findViewById(R.id.*textFilme*) // Onde estamos passando a referencia do objeto filme  
  
 */\*\* Bundle é um objeto do android que significa (PACOTE - EMBRULHAR)  
 Essa classe é utilizada para armazenar valores.  
 Esse objeto é uma maneira do android empacotar as informações e passar de uma tela para outra. \*/  
  
 /\*\* intent = Quando passamos uma intent(intenção) na tela anterior o proprio android studio ja passou esses dados e deixou guardado  
 \* so vamos precisar acessar eles agora  
 \*  
 \* extras = lembra que usamos o extra também? os dados estao dentro de inputExtra, vamos acessar eles também.  
 \*  
 \* agora o bundle tem todos os metodos passados da tela anterior pra ca.  
 \*/* val bundle = *intent*.*extras  
 /\*\*  
 \* Agora vamos recuperar o filme, ja passamos ele para dentro do bundle  
 \* Repara que estamos utilizando o getString porque o dado que passamos é do tipo String, mas poderia ser outro tipo tambem  
 \* como getInt para numeros inteiros.  
 \* vamos passar outros parametros para teste  
 \*  
 \* primeiro parametro é o nome que definimos la (key)  
 \*  
 \* Atenção: Devemos utilizar uma chamada segura (?) porque o objeto que esta vindo pode ser nulo.  
 \* bundle?.getString("filme")  
 \* Mas podemos também verificar se é diferente de nulo  
 \*  
 \* if (bundle != null) {  
 \* /\*\* Os dados podem vir nulo, por isso vamos verificar antes. \*/  
 \* bundle.getString("filme")  
 \* }  
 \*  
 \*/* if (bundle != null) {  
 */\*\* Os dados podem vir nulo, por isso vamos verificar antes. \*/* val filme = bundle.getString("filme")  
 textFilme.*text* = filme  
 }